

RECEIVED
CENTRAL FAX CENTER
OCT 24 2006

IN THE CLAIMS

Please amend claim 15 and cancel claims 1-4, 6-7, 11-14 and 16 (without prejudice to including them in a later-filed continuation application), as follows:

Listing of Claims:

1-7. (Cancelled)

8. (Previously Amended) A computer-implemented method, comprising:
creating one or more data structures that together store characteristics of a plurality of active branches B^{active} that make up a graph of nodes and branches that form a circuit, wherein B^{active} consists of

a set B^L of zero or more inductive branches, each having a non-zero inductive component but neither a capacitive component nor a variable switch state;

a set B^C of zero or more capacitive branches, each having a non-zero capacitive component but neither an inductive component nor a variable switch state; and

a set B^A of additional branches, each having neither an inductive component, nor a capacitive component;

partitioning B^{active} into a first branch set B_{tree}^{active} and a second branch set B_{link}^{active} , where the branches in B_{tree}^{active} form a spanning tree over B^{active} , giving priority in said partitioning to branches not in B^L over branches in B^L ;

sub-partitioning B_{link}^{active} into a third branch set B_{link}^L and a fourth branch set B_{link}^{CA} , where

$$B_{link}^L = B_{link}^{active} \cap B^L;$$

identifying a fifth branch set B^{CA} as the union of

$$B_{link}^{CA},$$

$$B^C \cap B_{tree}^{active}, \text{ and}$$

those branches in B_{tree}^{active} that form a closed graph when combined with B_{link}^{CA} ;

partitioning B^{CA} into a sixth branch set \tilde{B}_{tree}^{CA} and a seventh branch set \tilde{B}_{link}^{CA} , where the branches in \tilde{B}_{tree}^{CA} form a spanning tree over B^{CA} , giving priority in said partitioning to branches in B^C over branches not in B^C ;

identifying an eighth branch set $B_{tree}^C = \tilde{B}_{tree}^{CA} \cap B^C$;

selecting a set of state variables comprising:

for each branch of B_{link}^L , either the inductor current or inductor flux, and

for each branch of B_{tree}^C , either the capacitor voltage or capacitor charge; and

simulating a plurality of states of the circuit using the set of state variables.

9. (Original) The method of claim 8, wherein said partitioning steps each comprise an application of a weighted spanning tree algorithm.

10. (Original) The method of claim 9 wherein, for some positive numbers w_L and w_C : for the partitioning of B^{active} , a minimum spanning tree algorithm is used with weight

$$\text{function } \omega_L(b_j) = \begin{cases} w_L & \text{if branch } b_j \in B^L \\ 0 & \text{otherwise} \end{cases}; \text{ and}$$

for the partitioning of B^{CA} , a maximum spanning tree algorithm is used with weight

$$\text{function } \omega_C(b_j) = \begin{cases} w_C & \text{if branch } b_j \in B^C \\ 0 & \text{otherwise} \end{cases}$$

11-14. (Cancelled)

15. (Presently Amended) A system, comprising a processor and a computer-readable medium in communication with said processor, said medium containing programming instructions executable by said processor to:

build state equations for a first topology of an electronic circuit having at least two switching elements, wherein each switching element has a switching state;

solve said state equations at time t_i to provide a state output vector, in which at least two elements control the switching states of the switching elements;

calculate the value of a switching variable as a function of the state output vector, wherein the value reflects whether the switching state of at least one of the switching elements is changing; and

if the value of the switching variable at time t_i indicates that at least one of the switching elements is changing, determine a second topology of the electronic circuit for time t_i^+ and obtain state equations for the second topology;

wherein:

said programming instructions comprise a state equation building module, a solver module for ordinary differential equations, and a switching logic module;

said building is performed by the state equation building module;

said solving and calculating are performed by the solver module;

said determining is performed by the switching logic module;

The system of claim 12, wherein:

at a time t_i , at least two switching elements are each either rising-sensitive or falling-sensitive switches, wherein

rising-sensitive switches change switching state if and only if a controlling element of the state vector has passed from a negative value to a non-negative value; and

falling-sensitive switches change switching state if and only if a controlling element of the state vector has passed from a positive value to a non-positive value; and
the function is the arithmetic maximum of
a maximum of all controlling elements of the state vector that control rising-sensitive switches, and
the negative of the minimum of all controlling elements of the state vector that control falling-sensitive switches.

16. (Cancelled)